```
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO   
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO   
MMM         MMM     AAAAAAAAA        CCCCCCCCCCCC  RRRRRRRRRRRR        OOOOOOOOO   
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMMMMM   MMMMMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMM   MMM   MMM  AAA         AAA  CCC              RRR         RRR  OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRRRRRRRRRRR     OOO         OOO
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        OOO         OOO
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        OOO         OOO
MMM         MMM  AAAAAAAAAAAAAAA  CCC              RRR   RRR        OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRR      RRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRR      RRR     OOO         OOO
MMM         MMM  AAA         AAA  CCC              RRR      RRR     OOO         OOO
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO   
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO   
MMM         MMM  AAA         AAA     CCCCCCCCCCCC  RRR         RRR     OOOOOOOOO   
```

B 3

FILEID**ACTPRI

```
  AAAAAA      CCCCCCCC  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      IIIIII
  AAAAAA      CCCCCCCC  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      IIIIII
AA      AA  CC              TT      PP      PP  RR      RR      II
AA      AA  CC              TT      PP      PP  RR      RR      II
AA      AA  CC              TT      PP      PP  RR      RR      II
AA      AA  CC              TT      PP      PP  RR      RR      II
AA      AA  CC              TT      PPPPPPPP    RRRRRRRR        II
AA      AA  CC              TT      PPPPPPPP    RRRRRRRR        II
AAAAAAAAAA  CC              TT      PP          RR  RR          II
AAAAAAAAAA  CC              TT      PP          RR  RR          II
AA      AA  CC              TT      PP          RR    RR        II         ....
AA      AA  CC              TT      PP          RR    RR        II         ....
AA      AA    CCCCCCCC      TT      PP          RR      RR    IIIIII       ....
AA      AA    CCCCCCCC      TT      PP          RR      RR    IIIIII       ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

MA
VC

Table of contents

```
    (2)        85        DECLARATIONS
    (3)       137        PRMUN   PRIMARY UNARY OPERATORS
    (4)       193        PRMSYM  PRIMARY SYMBOLS
    (5)       260        NUMERIC PRIMARIES
    (6)       317        PROGRAM COUNTER PRIMARY
    (7)       351        ENTRY POINT MASK ROUTINES
    (8)       420        EXPRESSIONS
    (9)       487        UP-ARROW-A ASCII TEXT PRIMARY
    (10)      549        RADIX CONTROL
    (11)      585        OPERATORS
    (12)      632        SYMBOL ATTRIBUTE DIRECTIVES -GLOBL/DEBUG/WEAK/EXTRN
```

```
0000            .TITLE  MAC$ACTPRI PRIMARIES
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000    ; modules for input to the VAX-11 LINKER.
0000    ;
0000    ; ENVIRONMENT:  USER MODE
0000    ;
0000    ; AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03.01  MTR0013         Mike Rhodes     07-Jun-1982
0000    ;               Modify routine EXPBIN to test the Absolute
0000    ;               Expression flag MAC$GL_ABSFLAG a little closer
0000    ;               in order to interpret the expression type correctly.
0000    ;
0000    ;       V03.00  MTR0001         Mike Rhodes     15-Mar-1982
0000    ;               Modify routine NUMASC to use FLG$V_DLIMSTR flag
0000    ;               to allow passing hyphens and semicolons.
0000    ;               Fixes SPR #11-42904.
0000    ;
0000    ;       V02.12  PCG0008         Peter George    28-Aug-1981
0000    ;               Fix test for floating negation in PRMUN.
0000    ;
0000    ;       V02.11  PCG0002         Peter George    05-May-1981
0000    ;               Set SYM$M_RELPSECT flag in IDLIST and PRMSYM.
```

```
0000    ;
0000    ;       V01.10  RN0023          R. Newland      3-Nov-1979
0000    ;               New message codes to get error message from system
0000    ;               message file.
0000    ;
0000    ;       V01.09  RN0014          R. Newland      17-Oct-1979
0000    ;               Support for G_floating, H_floating, and Octaword data types.
0000    ;
0000    ;       V01.07  RN0005          R. Newland      12-Aug-1979
0000    ;               Remove .ALIGN LONG statements
0000    ;
0000    ;       V01.11  RN0027          R. Newland      14-Jan-1980
0000    ;               Fix problems with negative floating point literals.
0000    ;               SPR 11-27884.
0000    ;
0000    ;       V01.08  RN0007          R. Newland      28-Aug-1979
0000    ;               Fix problem with quadword ^A literals less
0000    ;               than 8 characters.  SPR 11-25674.
0000    ;
0000    ;       V01.05  0003            B. Schreiber    10-JAN-1979
0000    ;               Catch syntax error if pound sign forgotten before
0000    ;               ASCII immediate (^A) in operands.
0000    ;       V01.06  0006            B. Schreiber    16-JAN-1979
0000    ;               Fix problem with data generation if repeated data
0000    ;               and uparrow-A data (i.e. .BYTE ^A/ /[10])
0000    ;--
```

```
          0000            .SBTTL  DECLARATIONS
          0000    ;
          0000    ; INCLUDE FILES:
          0000    ;
          0000
          0000    ;
          0000    ; MACROS:
          0000    ;
          0000
          0000    $MAC_SYMBLKDEF                                     ;DEFINE SYMBOL BLOCK OFFSETS
          0000    $MAC_CTLFLGDEF                                     ;DEFINE CONTROL FLAGS
          0000    $MAC_INTCODDEF                                     ;DEFINE INT. FILE COMMANDS
          0000    $MAC_GENVALDEF                                     ;DEFINE OTHER GOOD SYMBOLS
          0000    $MACMSGDEF                                         ; Define message codes
          0000
          0000    ;
          0000    ; EQUATED SYMBOLS:
          0000    ;
          0000
80000000  0000            SIGN_BIT        =       ^X80000000      ;SIGN BIT
          0000
          0000    ;
          0000    ; OWN STORAGE:
          0000    ;
          0000
      00000000            .PSECT  MAC$RO_DATA,NOEXE,NOWRT,GBL,LONG
          0000
          0000    ;++
          0000    ;       THIS DISPATCH TABLE IS USED DURING PASS 1 TO JSB TO
          0000    ;       MATH ACTION ROUTINES.
          0000    ;--
          0000
          0000    P1$ARITH_DISP::
00000000  0000            .LONG   0                                  ;(0)--SHOULD NOT HAPPEN
00000000' 0004            .LONG   P1$ARITH_ADD                       ;INT$_ADD
00000000' 0008            .LONG   P1$ARITH_AND                       ;INT$_AND
00000000' 000C            .LONG   P1$ARITH_ASH                       ;INT$_ASH
00000000' 0010            .LONG   P1$ARITH_DIV                       ;INT$_DIV
00000000' 0014            .LONG   P1$ARITH_MUL                       ;INT$_MUL
00000000' 0018            .LONG   P1$ARITH_NEG                       ;INT$_NEG
00000000' 001C            .LONG   P1$ARITH_NOT                       ;INT$_NOT
00000000' 0020            .LONG   P1$ARITH_OR                        ;INT$_OR
00000000' 0024            .LONG   P1$ARITH_SAME                      ;INT$_SAME
00000000' 0028            .LONG   P1$ARITH_SUB                       ;INT$_SUB
00000000' 002C            .LONG   P1$ARITH_XOR                       ;INT$_XOR
          0030
      00000000            .PSECT  MAC$ACTPRI_DATA,NOEXE,LONG
          0000
    0000  0000    SYM_FLAG:.WORD  0                                  ;USED FOR GLOBAL/DEBUG/WEAK/EXTERN
          0002    ENTRY_MASK:
    0000  0002            .WORD   0                                  ;USED FOR .ENTRY/.VECTOR
```

```
                          0004            .SBTTL  PRMUN   PRIMARY UNARY OPERATORS
                          0004
                          0004    ;++
                          0004    ; FUNCTIONAL DESCRIPTION:
                          0004    ;
                          0004    ;       PRMUN IS CALLED WHEN A UNARY OPERATOR PRODUCTION IS
                          0004    ;       ENCOUNTERED.  IT CHECKS FOR UNARY FLOATING NEGATION
                          0004    ;       AND CHANGES IT TO AN XOR OF THE SIGN BIT.  IF THE
                          0004    ;       EXPRESSION IS TO BE EVALUATED IN PASS 2 THE INTERMEDIATE
                          0004    ;       CODE IS EMITTED.  THE EXPRESSION IS THEN EVALUATED
                          0004    ;       AND THE RESULT IS RETURNED IN MAC$GL_VALUE.
                          0004    ;
                          0004    ; INPUTS:
                          0004    ;
                          0004    ;       MAC$AL_VALSTACK[R7]     VALUE OF EXPRESSION
                          0004    ;       MAC$AL_VALSTACK-4[R7]   OPERATION
                          0004    ;
                          0004    ; OUTPUTS:
                          0004    ;
                          0004    ;       MAC$GL_VALUE            COMPUTED VALUE
                          0004    ;
                          0004    ;--
                          0004
                      00000000            .PSECT  MAC$RO_CODE_P1,NOWRT,GBL,LONG
                          0000
                          0000            .ENABL  LSB
                          0000
                          0000    PRMUN::                                 ;PRIMITIVE = OPUNARY PRIMITIVE
56    0000'CF47    DE     0000            MOVAL   W^MAC$AL_VALSTACK[R7],R6;POINT TO TOP OF VALUE STACK
                          0006            $VPUSH  (R6)                    ;PUSH VALUE ONTO STACK
              76   D5     000E            TSTL    -(R6)                   ;BACK UP TO ROUTINE VALUE
                          0010    ;
                          0010    ; IF OPERATION IS FLOATING NEGATION, CHANGE TO XOR OF SIGN BIT
                          0010    ;
         06   66   91     0010            CMPB    (R6),#INT$_NEG          ;ARE WE DOING A NEGATE?
              26   12     0013            BNEQ    20$                     ;IF NEQ NO
       0000'CF     91     0015            CMPB    W^MAC$GB_RDXNDX,-       ;YES--AND IS IT A FLOATING NEGATE?
              04          0019                    #RDX$V_FLOAT            ;...
              1F   1F     001A            BLSSU   20$                     ;IF LESS NO
     0C 6B    06   E1     001C            BBC     #FLG$V_EVALEXPR,(R11),10$ ;YES--ARE WE EVALUATING ON PASS 2
                          0020            $INTOUT_LW INT$_STKL,<#^X8000>  ; Yes--stack floating point sign bit
         66   0B   9A     002C    10$:    MOVZBL  #INT$_XOR,(R6)          ;CHANGE COMMAND TO XOR
                          002F            $VPUSH  #^X8000                 ; Stack sign bit
         50   66   D0     003B    20$:    MOVL    (R6),R0                 ;GET ACTION
         56   50   D0     003E            MOVL    R0,R6                   ;REMEMBER IT FOR LATER
     0B 6B    06   E1     0041            BBC     #FLG$V_EVALEXPR,(R11),30$ ;BRANCH IF NOT EVAL ON PASS 2
         51   02   9A     0045            MOVZBL  #2,R1                   ;SET BYTE COUNT
           FFB5'   30     0048            BSBW    MAC$INTOUT_X            ;EMIT TO INT. FILE
    0000'CF   59   D0     004B            MOVL    R9,W^MAC$GL_EXPEND      ;SAVE END OF EXPR POINTER
56    0000'CF46    D0     0050    30$:    MOVL    W^P1$ARITH_DISP[R6],R6  ;GET ADDRESS OF ROUTINE
              66   16     0056            JSB     (R6)                    ;CALL IT
                          0058            $VPOP   W^MAC$GL_VALUE          ;RETRIEVE VALUE
                   05     0062            RSB
                          0063
                          0063            .DSABL  LSB
```

```
                              0063          .SBTTL  PRMSYM  PRIMARY SYMBOLS
                              0063
                              0063  ;++
                              0063  ; FUNCTIONAL DESCRIPTION
                              0063  ;
                              0063  ;       PRMSYM IS INVOKED WHEN AN ID IS FOUND IN THE PRODUCTION.
                              0063  ;       BASED ON THE SYMBOL ATTRIBUTES (LOCAL, GLOBAL, EXTERNAL,
                              0063  ;       DEFINED, ABSOLUTE) IT WILL SET CONTROL FLAGS FOR LATER
                              0063  ;       PROCESSING OF THE ID.
                              0063  ;
                              0063  ; INPUTS:
                              0063  ;
                              0063  ;       MAC$GL_VALUE            POINTER TO ID SYMBOL BLOCK
                              0063  ;
                              0063  ; OUTPUTS:
                              0063  ;
                              0063  ;--
                              0063
                              0063  PRMSYM::                                        ;PRIMARY = ID
                              0063
         56   0000'CF     D0  0063          MOVL    W^MAC$GL_VALUE,R6       ;GET POINTER TO SYMBOL BLOCK
           18 6B     18   E0  0068          BBS     #FLG$V_NOREF,(R11),5$   ;BRANCH IF WE SHOULD NOT REF SYMBOL
      09 A6   0080 8F     A8  006C          BISW2   #SYM$M_REF,SYM$W_FLAG(R6) ;FLAG SYMBOL AS REFERENCED
      50   00000000'EF    D0  0072          MOVL    MAC$GL_PSECTPTR,R0      ;GET POINTER TO PSECT DATA
        06 0D A0     03   E1  0079          BBC     #PSC$V_REL, -           ;IF ABS PSECT
                              007E                  PSC$W_OPTIONS(R0),5$    ;THEN SKIP
      09 A6   0800 8F     A8  007E          BISW2   #SYM$M_RELPSECT,SYM$W_FLAG(R6)   ;SET REL PSECT FLAG
      09 A6   4000 8F     AA  0084  5$:     BICW2   #SYM$M_SUPR,SYM$W_FLAG(R6) ;AND CLEAR SUPPRESS BIT
        03 09 A6     00   E0  008A          BBS     #SYM$V_DEF,SYM$W_FLAG(R6),10$
                              008F                                          ;IF SYMBOL NOT YET DEFINED
              6B     04   CA  008F          BICL2   #FLG$M_COMPEXPR,(R11)   ;THEN EXPR VALUE NOT YET KNOWN
                     0C   B3  0092  10$:    BITW    #SYM$M_GLOBL!SYM$M_EXTRN,- ;SYMBOL GLOBAL OR EXTERNAL?
                  09 A6       0094                      SYM$W_FLAG(R6)
                     1B   13  0096          BEQL    20$                     ;IF EQL NO
              0041 8F     B3  0098          BITW    #SYM$M_DEF!SYM$M_LOCAL,-;YES--DEFINED OR LOCAL?
                  09 A6       009C                      SYM$W_FLAG(R6)      ;
                     13   12  009E          BNEQ    20$                     ;IF NEQ NO
                              00A0                                          ;
                              00A0                                          ;SYMBOL IS EXTERNAL OR GLOBAL
                              00A0                                          ; SYMBOL NOT YET DEFINED
           3C 6B     06   E1  00A0          BBC     #FLG$V_EVALEXPR,(R11),50$ ;EVALUATE ON PASS 2?
                              00A4          $INTOUT_LW INT$_STKG,R6         ;YES--STACK GLOBAL
        0000'CF      59   D0  00AC          MOVL    R9,W^MAC$GL_EXPEND      ;SAVE END OF EXPRESSION
                     2D   11  00B1          BRB     50$
                              00B3  ;
                              00B3  ; LOCAL OR DEFINE SYMBOL
                              00B3  ;
     0000'CF      0C A6   91  00B3  20$:    CMPB    SYM$B_SEG(R6),W^MAC$GL_PRMSEG ;DIFFERENT PSECTS?
                     0E   13  00B9          BEQL    30$                     ;IF EQL NO
        09 09 A6     04   E0  00BB          BBS     #SYM$V_ABS,SYM$W_FLAG(R6),30$ ;YES--UNLESS SYMBOL ABSOLUTE
                              00C0                                          ;(BRANCH IF ABSOLUTE)
             0000'CF      D5  00C0          TSTL    W^MAC$GL_PRMSEG         ;REALLY DIFFERENT PSECTS?
                     03   13  00C4          BEQL    30$                     ;IF EQL NO
              6B     04   CA  00C6          BICL2   #FLG$M_COMPEXPR,(R11)   ;YES--VALUE NOT YET KNOWN
           0D 6B     06   E1  00C9  30$:    BBC     #FLG$V_EVALEXPR,(R11),40$ ;EVALUATE ON PASS 2?
                              00CD          $INTOUT_LW INT$_STKS,R6         ;YES--STACK SYMBOL
        0000'CF      59   D0  00D5          MOVL    R9,W^MAC$GL_EXPEND      ;SAVE END OF EXPRESSION
```

```
0000'CF    05 A6   D0  00DA   250 40$:    MOVL    SYM$L_VAL(R6),W^MAC$GL_VALUE ;VALUE IS VALUE OF SYMBOL
  04 09 A6    04   E1  00E0   251 50$:    BBC     #SYM$V_ABS,SYM$W_FLAG(R6),60$ ;IS SYMBOL ABSOLUTE?
     10 6B    02   E0  00E5   252         BBS     #FLG$V_COMPEXPR,(R11),70$ ;YES--DO WE KNOW EXPR VALUE?
       0000'CF     D6  00E9   253 60$:    INCL    W^MAC$GL_ABSFLAG          ;NO--NOT ABSOLUTE EXPRESSION
       0000'CF     D5  00ED   254         TSTL    W^MAC$GL_PRMSEG           ;DOES EXPR HAVE A SEG YET?
            06     12  00F1   255         BNEQ    70$                       ;IF NEQ YES
0000'CF    0C A6   9A  00F3   256         MOVZBL  SYM$B_SEG(R6),W^MAC$GL_PRMSEG ;NO--USE SYMBOL SEGMENT
     55    00'8F   9A  00F9   257 70$:    MOVZBL  #CRF$K_REF,R5             ;SET REFERENCE
          FF00'    31  00FD   258         BRW     MAC$CREF_SYM              ;CREF SYMBOL IF CREFFING AND RETURN
```

```
                           0100         .SBTTL  NUMERIC PRIMARIES
                           0100
                           0100 ;++
                           0100 ; FUNCTIONAL DESCRIPTION:
                           0100 ;
                           0100 ;       NUMFLT IS CALLED WHEN '^F' IS SEEN. A FLOATING POINT NUMBER
                           0100 ;       IS SCANNED.
                           0100 ;
                           0100 ;--
                           0100
                           0100 NUMFLT::                                    ;SPECIAL_OPERATOR = DUPF
               FEFD'   30  0100         BSBW    MAC$SKIPSP                  ;SKIP SPACES
               FEFA'   30  0103         BSBW    MAC$GETFLOAT                ;ACCUMULATE FLOATING POINT NUMBER
                 00    11  0106         BRB     PRMINT                      ;TREAT AS INTEGER
                           0108
                           0108 ;++
                           0108 ; FUNCTIONAL DESCRIPTION:
                           0108 ;
                           0108 ;       PRMINT IS CALLED WHEN AN INTEGER (OR INTEGER-LIKE) TOKEN
                           0108 ;       IS FOUND.  IF THE EXPRESSION IS BEING EVALUATED IN PASS 2
                           0108 ;       THE VALUE IS EMITTED TO THE INTERMEDIATE FILE.
                           0108 ;
                           0108 ;--
                           0108
                           0108 PRMINT::                                    ;PRIMARY = DINTEGER
        0F 6B    06    E1  0108         BBC     #FLG$V_EVALEXPR,(R11),10$ ;EVALUATE ON PASS 2?
                           010C         $INTOUT_LW INT$_STKL,<W^MAC$GL_VALUE> ;YES--STACK VALUE
     0000'CF     59    D0  0116         MOVL    R9,W^MAC$GL_EXPEND          ;SAVE END OF EXPRESSION
                       05  011B 10$:    RSB
                           011C
                           011C ;++
                           011C ; FUNCTIONAL DESCRIPTION:
                           011C ;
                           011C ;       PRMBRK IS CALLED WHEN AN EXPRESSION IN ANGLE BRACKETS IS
                           011C ;       SCANNED.  THE VALUE IS PICKED OFF OF THE STACK AND PLACED
                           011C ;       IN MAC$GL_VALUE.
                           011C ;
                           011C ;--
                           011C
                           011C PRMBRK::                                    ;PRIMARY = DANGOPN EXPR DANGCLS
  0000'CF    FFFC'CF47     D0  011C         MOVL    W^MAC$AL_VALSTACK-4[R7],- ;VALUE IS ON STACK
                           0124                     @^MAC$GL_VALUE  ;
                       05  0124         RSB
                           0125
                           0125 ;++
                           0125 ; FUNCTIONAL DESCRIPTION:
                           0125 ;
                           0125 ;       PRMRDX IS CALLED WHEN A RADIX CONTROL PRIMARY HAS BEEN
                           0125 ;       SCANNED.  THE RADIX IS RESET TO THE PREVIOUS RADIX.
                           0125 ;
                           0125 ;--
                           0125
                           0125 PRMRDX::                                    ;PRIMARY = RADIX CONTROL PRIMARY
  0000'CF    FFFC'CF47     F6  0125         CVTLB   W^MAC$AL_VALSTACK-4[R7],- ;RESET TO PREVIOUS
                           012D                     @^MAC$GB_RDXNDX    ;RADIX
                       05  012D         RSB
```

```
                             012E          .SBTTL  PROGRAM COUNTER PRIMARY
                             012E
                             012E  ;++
                             012E  ; FUNCTIONAL DESCRIPTION:
                             012E  ;
                             012E  ;       PRMPC IS CALLED WHEN A PC REFERENCE (".") IS SCANNED.
                             012E  ;       IF THE EXPRESSION IS BEING EVALUATED ON PASS 2 THE
                             012E  ;       CODE IS EMITTED TO STACK THE PC VALUE.  IF THE EXPRESSION
                             012E  ;       CONTAINS CROSS-PSECT REFERENCES THEN THE EXPRESSION IS
                             012E  ;       NOT A COMPILE-TIME EXPRESSION AND THE FLAG (FLG$M_COMPEXPR)
                             012E  ;       IS CLEARED IN THE FLAGS WORD.
                             012E  ;
                             012E  ;--
                             012E
                             012E  PRMPC::                                        ;PRIMARY = DPC
        0B 6B    06     E1   012E          BBC     #FLG$V_EVALEXPR,(R11),10$ ;BR IF NOT EVALUATE ON PASS 2
                             0132          $INTOUT_X INT$_STKPC              ;YES--STACK PC
      0000'CF    59     D0   0138          MOVL    R9,W^MAC$GL_EXPEND        ;SAVE END OF EXPRESSION
 0000'CF    0000'CF     D0   013D  10$:    MOVL    W^MAC$GL_PC,W^MAC$GL_VALUE ;RETURN VALUE IS PC
      50    0000'CF     D0   0144          MOVL    W^MAC$GL_PSECTPTR,R0      ;GET POINTER TO CURRENT PSECT
     1F 0D A0    03     E1   0149          BBC     #PSC$V_REL,PSC$W_OPTIONS(R0),30$
                             014E                                            ;BRANCH IF ABS PSECT
            0000'CF     D6   014E          INCL    W^MAC$GL_ABSFLAG          ;RELOCATABLE--FLAG NOT ABS EXPR
            0000'CF     D5   0152          TSTL    W^MAC$GL_PRMSEG           ;EXPR HAVE A PSECT YET?
                 09     12   0156          BNEQ    20$                       ;IF NEQ YES
            0000'CF     D0   0158          MOVL    W^MAC$GL_PSECT,-          ;NO--USE CURRENT PSECT
            0000'CF          015C                  @^MAC$GL_PRMSEG
                 0C     11   015F          BRB     30$
            0000'CF     D1   0161  20$:    CMPL    W^MAC$GL_PRMSEG,-         ;YES--CROSS PSECT REFERENCES?
            0000'CF          0165                  @^MAC$GL_PSECT
                 03     13   0168          BEQL    30$                       ;IF EQL NO
            6B   04     CA   016A          BICL2   #FLG$M_COMPEXPR,(R11)     ;YES--FLAG NOT COMPILE EXPRESSION
                        05   016D  30$:    RSB
                        016E            .SBTTL  ENTRY POINT MASK ROUTINES
                        016E
                        016E    ;++
                        016E    ; FUNCTIONAL DESCRIPTION:
                        016E    ;
                        016E    ;       RGLST1 AND REGLST ARE CALLED TO ACCUMULATE AN ENTRY-POINT
                        016E    ;       MASK.  RGLST1 IS CALLED FOR THE FIRST ITEM TO INITIALIZE THE
                        016E    ;       ENTRY MASK TO ZERO, AND REGLST IS CALLED FOR EACH SUCCESSIVE
                        016E    ;       ITEM IN THE MASK.  THE APPROPRIATE BIT IN ENTRY_MASK IS
                        016E    ;       SET FOR LATER PROCESSING BY THE 'MASK' ROUTINE.
                        016E    ;
                        016E    ;--
                        016E
                        016E    RGLST1::                                        ;REGLIS = MASK_ITEM
       0002'CF    B4    016E            CLRW    W^ENTRY_MASK                    ;START WITH 0
                        0172    REGLST::                                        ;REGLIS = REGLIS MASK_ITEM
  50   0000'CF47  D0    0172            MOVL    W^MAC$AL_VALSTACK[R7],R0;GET THE MASK BIT NUMBER
00 0002'CF    50  E3    0178            BBCS    R0,W^ENTRY_MASK,10$     ;SET THE BIT IN THE MASK
                  05    017E    10$:    RSB
                        017F
                        017F    ;++
                        017F    ; FUNCTIONAL DESCRIPTION:
                        017F    ;
                        017F    ;       MASK IS CALLED WHEN AN ENTRY-POINT MASK HAS BEEN ACCUMULATED
                        017F    ;       IN ENTRY_MASK.  IF WE ARE EVALUATEING EXPRESSIONS THE VALUE
                        017F    ;       WILL BE STACKED IN PASS 2.
                        017F    ;
                        017F    ;--
                        017F
                        017F            .ENABL  LSB
                        017F
                        017F    MASK::                                          ;REGISTER_MASK = DUPM DANGOPN REGLIS DANGCLS
  50   0002'CF    3C    017F            MOVZWL  W^ENTRY_MASK,R0                 ;PICK UP MASK WORD
              10  11    0184            BRB     10$                             ;FINISH IN COMMON CODE
                        0186
                        0186    ;++
                        0186    ; FUNCTIONAL DESCRIPTION:
                        0186    ;
                        0186    ;       MASKX IS CALLED WHEN '^MRn' IS SCANNED.  A MASK IS CREATED
                        0186    ;       AND THE VALUE IS SENT TO PASS 2 IF EXPRESSIONS ARE BEING
                        0186    ;       EVALUATED.
                        0186    ;
                        0186    ;--
                        0186
                        0186    MASKX::                                         ;REGISTER_MASK = DUPM MASK_ITEM
  51   0000'CF47  D0    0186            MOVL    W^MAC$AL_VALSTACK[R7],R1;GET MASK BIT NUMBER
              50  D4    018C            CLRL    R0                              ;START WITH A CLEAN SLATE
      04 50   51  E3    018E            BBCS    R1,R0,10$                       ;SET THE MASK BIT AND JOIN COMMON CODE
              02  11    0192            BRB     10$                             ;BETTER SAFE THAN SORRY
                        0194
                        0194    ;++
                        0194    ; FUNCTIONAL DESCRIPTION:
                        0194    ;
                        0194    ;       MASKNL IS CALLED WHEN A NULL ENTRY-MASK IS SCANNED.  IF
                        0194    ;       EXPRESSSIONS ARE BEING EVALUATED, A ZERO  IS STACKED IN
                        0194    ;       PASS 2.
                        0194    ;
```

```
                          0194  ;--
                          0194
                          0194  MASKNL::                                                ;REGISTER MASK = DUPM DANGOPN DANGCLS
                  50  D4  0194          CLRL    R0                                      ;RESULT IS 0
        0000'CF   50  D0  0196  10$:    MOVL    R0,W^MAC$GL_VALUE                       ;STORE RESULT
          0F 6B   06  E1  019B          BBC     #FLG$V_EVALEXPR,(R11),20$ ;BRANCH IF NO EXPRESSION EVALUATION
                          019F          $INTOUT_LW INT$_STKL,<W^MAC$GL_VALUE> ;YES--SEND VALUE TO PASS 2
        0000'CF   59  D0  01A9          MOVL    R9,W^MAC$GL_EXPEND                      ;SAVE END OF EXPRESSION
                      05  01AE  20$:    RSB
                          01AF
                          01AF          .DSABL  LSB
```

```
                             01AF                 .SBTTL  EXPRESSIONS
                             01AF
                             01AF      ;++
                             01AF      ;
                             01AF      ;          VAX-11 MACRO RECOGNIZES DIFFERENT TYPES OF EXPRESSIONS. THESE
                             01AF      ;          ROUTINES PROCESS COMPILE-TIME EXPRESSIONS.  THE RESULT OF SUCH
                             01AF      ;          AN EXPRESSION IS A LONGWORD WHICH WILL BE KNOWN BY PASS2
                             01AF      ;          (OR AT LINK TIME IF THE EXPRESSION INVOLVES GLOBALS OR
                             01AF      ;          EXTERNALS).  THE MOST COMMON USAGE OF THIS TYPE OF EXPRESSIONS
                             01AF      ;          IS IN OPERANDS.   ANOTHER TYPE OF EXPRESSION IS FOUND IN THE
                             01AF      ;          ASSIGNMENT STATMENT WHERE AN EXPRESSION GENERATES CODE TO
                             01AF      ;          EVALUATE THE EXPRESSION AT RUN TIME.
                             01AF      ;
                             01AF      ;          THE 'PRIMITIVE' ROUTINES SET FLAGS DESCRIBING THE EXPRESSION.
                             01AF      ;          THESE FLAGS MUST BE INITIALIZED BY THE EXPRESSION CALLER IF
                             01AF      ;          THEY ARE TO BE USED.
                             01AF      ;
                             01AF      ;          FLG$M_COMPEXPR  FALSE IF EXPRESSION VALUE NOT YET KNOWN
                             01AF      ;          FLG$M_EVALEXPR  TRUE CAUSES EVALUATION TO OCCUR ON PASS 2
                             01AF      ;          FLG$M_ABSEXPR   TRUE INDICATES THAT EXPRESSION IS ABSOLUTE
                             01AF      ;
                             01AF      ;--
                             01AF
                             01AF      EXPBIN::                                       ;EXPR = EXPR OPBINARY PRIMARY
          55     57     D0   01AF                 MOVL    R7,R5                       ;COPY STACK POINTER
                             01B2                 $VPUSH  W^MAC$AL_VALSTACK-8[R5]     ;PUSH LEFT OPERAND ONTO STACK
                             01BD                 $VPUSH  W^MAC$AL_VALSTACK[R5]       ;PUSH RIGHT OPERAND
   56   FFFC'CF45       D0   01C8                 MOVL    W^MAC$AL_VALSTACK-4[R5],R6  ;GET COMMAND FROM STACK
      0B 6B      06     E1   01CE                 BBC     #FLG$V_EVALEXPR,(R11),10$   ;EVALUEATE ON PASS 2?
          50     56     D0   01D2                 MOVL    R6,R0                       ;YES--GET COMMAND
               FE28'    30   01D5                 BSBW    MAC$INTOUT_X                ;OUTPUT CMD TO INT FILE
     0000'CF     59     D0   01D8                 MOVL    R9,W^MAC$GL_EXPEND          ;SAVE END OF EXPRESSSION PTR
                 56     D5   01DD      10$:       TSTL    R6                          ;WAS ROUTINE SUPPLIED?
                 58     13   01DF                 BEQL    40$                         ;IF EQL NO
          0A     56     91   01E1                 CMPB    R6,#INT$_SUB                ;SUBTRACTION?
                 10     12   01E4                 BNEQ    20$                         ;IF NEQ NO
     01     0000'CF     D1   01E6                 CMPL    W^MAC$GL_ABSFLAG,#1         ;YES--SEVERAL RELATIVE REFS?
                 09     15   01EB                 BLEQ    20$                         ;IF LEQ NO
      05 6B      02     E1   01ED                 BBC     #FLG$V_COMPEXPR,(R11),20$   ;YES--REALLY COMPILE TIME EXPR?
     0000'CF     02     C2   01F1                 SUBL2   #2,W^MAC$GL_ABSFLAG         ;YES--MAKE RESULT ABSOLUTE
         0000'CF        D4   01F6      20$:       CLRL    W^MAC$GL_VAL3               ;CLEAR EXPRESSION OVERFLOWW IND.
                 56     DD   01FA                 PUSHL   R6                          ;SAVE ROUTINE IDENT.
   56   0000'CF46       D0   01FC                 MOVL    W^P1$ARITH_DISP[R6],R6      ;GET ROUTINE ADDRESS
                 66     16   0202                 JSB     (R6)                        ;CALL ROUTINE
                 56   8ED0   0204                 POPL    R6                          ;RESTORE ROUTINE IDENT
     51     0000'CF     D0   0207                 MOVL    W^MAC$GL_VAL3,R1            ;EXPRESSION OVERFLOW?
                 33     13   020C                 BEQL    50$                         ;IF EQL NO
         0000'CF        D5   020E                 TSTL    W^MAC$GL_ABSFLAG            ;YES--ABSOLUTE EXPRESSION?
                 2D     12   0212                 BNEQ    50$                         ;IF NEQ NO
 52   007D8810 8F       D0   0214                 MOVL    #MAC$_EXPOVR32,R2           ; No--assume expression overflow
          04     56     91   021B                 CMPB    R6,#INT$_DIV                ;UNLESS IT WAS DIVISION
                 0B     12   021E                 BNEQ    30$                         ;IF NEQ NO
                 51     D5   0220                 TSTL    R1                          ;THEN CHECK FOR DIVIDE BY 0
                 07     18   0222                 BGEQ    30$                         ;IF GEQ THEN NOT DIVIDE BY 0
 52   007D8808 8F       D0   0224                 MOVL    #MAC$_DIVBYZERO,R2          ; It was divide by zero
                             022B      30$:       $INTOUT_LW INT$_WRN,<R2,W^MAC$GL_ERRPT> ;EMIT ERROR TO PASS 2
                 08     11   0237                 BRB     50$
                             0239      ;
```

```
                                   0239  ; NO ROUTINE SUPPLIED
                                   0239  ;
                                   0239  40$:    $VPUSH  #0                         ;RESULT IS 0
                                   0241  50$:    $VPOP   W^MAC$GL_VALUE             ;POP RESULT INTO MAC$GL_VALUE
              01    0000'CF   D1   024B          CMPL    W^MAC$GL_ABSFLAG,#1        ;SEVERAL RELATIVE REFERENCES?
                          0C  15   0250          BLEQ    60$                        ;IF LEQ NO
             05 00000000'GF   E9   0252          BLBC    G^MAC$GL_ABSFLAG,60$       ;YES -- ARE THE NUMBER OF REFs ODD?
                0000'CF   01  9A   0259          MOVZBL  #1,W^MAC$GL_ABSFLAG        ;YES -- CALL IT ONE (RESULT IS...
                              05   025E  60$:    RSB                                ;...THE EXPRESSION IS RELOCATEABLE)
```

```
                             025F        .SBTTL  UP-ARROW-A ASCII TEXT PRIMARY
                             025F
                             025F  ;++
                             025F  ; FUNCTIONAL DESCRIPTION:
                             025F  ;
                             025F  ;       NUMASC IS INVOKED WHEN THE PRODUCTION 'UP-ARROW-A' IS
                             025F  ;       FOUND IN THE INPUT.  IT SCANS THE NEXT CHARACTER AS A
                             025F  ;       DELIMITER, THEN READS TEXT, STORING UP TO THE MAXIMUM
                             025F  ;       NUMBER OF CHARACTERS IN 'MAC$GL_VALUE', LOOKING FOR
                             025F  ;       THE MATCHING DELIMITER.  IF THE MAXIMUM NUMBER OF BYTES
                             025F  ;       FOR THIS OPERAND IS EXCEEDED OR IF END-OF-LINE IS FOUND
                             025F  ;       BEFORE THE MATCHING DELIMITER, A MESSAGE IS OUTPUT
                             025F  ;       TO PASS 2.
                             025F  ;
                             025F  ;--
                             025F
                             025F  NUMASC::                                         ;SPECIAL OPERATOR = DUPA
         00 6B    24    E3   025F          BBCS    #FLG$V_UPAFLG,(R11),.+1        ;FLAG DUPA WAS SEEN
              FD9A'     30   0263          BSBW    MAC$SKIPSP                     ;SKIP SPACES AND TABS
      56   0000'CF      9E   0266          MOVAB   W^MAC$GQ_VALUEQ,R6             ;POINT TO RESULT AREA
                  66    7C   026B          CLRQ    (R6)                           ;CLEAR OUT 8 BYTES
               08 A6    7C   026D          CLRQ    8(R6)                          ; and then the next 8 bytes
           55     5A    D0   0270          MOVL    R10,R5                         ;COPY DELIMITER
           0D     55    91   0273          CMPB    R5,#CR                         ;IS DELIMITER CR?
                  1F    13   0276          BEQL    20$                            ;IF EQL YES--ERROR
      54   0000'CF      9A   0278          MOVZBL  W^MAC$GL_OPSIZE,R4             ;GET MAX SIZE OF OPERAND
         00 6B    2F    E3   027D          BBCS    #FLG$V_DLIMSTR,(R11),.+1 ; PASS ALL CHARACTERS (EVEN -;)
              FD7C'     30   0281  10$:    BSBW    MAC$GETCHR                     ;GET NEXT CHARACTER
           55     5A    91   0284          CMPB    R10,R5                         ;DELIMITER?
                  1C    13   0287          BEQL    30$                            ;IF EQL YES
           0D     5A    91   0289          CMPB    R10,#CR                        ;NO--END OF LINE?
                  09    13   028C          BEQL    20$                            ;IF EQL YES--ERROR
                  54    D7   028E          DECL    R4                             ;NO--ROOM TO STORE BYTE?
                  EF    19   0290          BLSS    10$                            ;DON'T STORE IF TOO MANY CHARS
           86     5A    90   0292          MOVB    R10,(R6)+                      ;STORE CHARACTER
                  EA    11   0295          BRB     10$                            ;LOOP FOR MORE
                             0297  ;
                             0297  ; FOUND EOL BEFORE DELIMITER
                             0297  ;
                             0297  20$:    $MAC_ERR UNTERMARG                     ; Get message code
              FD61'     30   029C          BSBW    MAC$ERRORPT                    ;ISSUE MESSAGE TO PASS 2
         09 6B    2F    E4   029F          BBSC    #FLG$V_DLIMSTR,(R11),40$ ;CLEAR ALLCHR AND GO FINISH UP
                  07    11   02A3          BRB     40$                            ;FINISH
                             02A5  ;
                             02A5  ; FOUND OTHER DELIMITER
                             02A5  ;
         00 6B    2F    E4   02A5  30$:    BBSC    #FLG$V_DLIMSTR,(R11),.+1 ;DO NOT PASS ALL CHARACTERS
              FD54'     30   02A9          BSBW    MAC$GETCHR                     ;SKIP OVER DELIMITER
                  54    D5   02AC  40$:    TSTL    R4                             ;TOO MANY CHARACTERS?
                  10    18   02AE          BGEQ    50$                            ;IF GEQ NO
                             02B0          $INTOUT_LW INT$_WRN,<#MAC$_DATATRUNC,W^MAC$GL_ERRPT> ; Yes--report error
                             02C0  50$:
      08   0000'CF      91   02C0          CMPB    W^MAC$GL_OPSIZE,#8             ; Was this a QUAD or OCTA operand?
                  15    19   02C5          BLSS    70$                            ; No if LSS
 0000'CF   0000'CF      D0   02C7          MOVL    W^MAC$GL_VAL3, -               ; Yes: save bits 32 to 63
                             02CE                  W^MAC$GL_HIGH_32
      10   0000'CF      91   02CE          CMPB    W^MAC$GL_OPSIZE,#16            ; Was this an OCTA operand?
```

```
                      07   12  02D3    544         BNEQ    70$                       ; No if NEQ
       0000'CF   0000'CF   7D  02D5    545         MOVQ    W^MAC$GQ_VAL2, -          ; Yes: save bits 64 to 127
                               02DC    546                 W^MAC$GQ_HIGH_64
                    FE29   31  02DC    547 70$:    BRW     PRMINT                    ;TREAT AS INTEGER DATA
```

```
                              02DF          .SBTTL  RADIX CONTROL
                              02DF
                              02DF  ;++
                              02DF  ; FUNCTIONAL DESCRIPTION:
                              02DF  ;
                              02DF  ;       THESE FOUR ROUTINES ARE INVOKED WHEN A RADIX CONTROL
                              02DF  ;       PRIMARY IS ENCOUNTERED.  THESE ROUTINES SAVE THE
                              02DF  ;       OLD RADIX IN MAC$GL_VALUE (FOR LATER RESTORATION) AND
                              02DF  ;       SET THE NEW RADIX IN MAC$GB_RDXNDX.
                              02DF  ;
                              02DF  ;--
                              02DF
                              02DF  RDXBIN::                                    ;RADIX_CONTROL = DUPB
                  0A    10    02DF          BSBB    SET_RADIX                   ;GO SET THE INDEX FOR BINARY
                        00    02E1          .BYTE   RDX$V_BINARY
                              02E2
                              02E2  RDXDEC::                                    ;RADIX_CONTROL = DUPD
                  07    10    02E2          BSBB    SET_RADIX                   ;SET DECIMAL RADIX
                        02    02E4          .BYTE   RDX$V_DECIMAL
                              02E5
                              02E5  RDXOCT::                                    ;RADIX_CONTROL = DUPO
                  04    10    02E5          BSBB    SET_RADIX                   ;SET OCTAL RADIX
                        01    02E7          .BYTE   RDX$V_OCTAL
                              02E8
                              02E8  RDXHEX::                                    ;RADIX_CONTROL = DUPX
                  01    10    02E8          BSBB    SET_RADIX                   ;SET HEX RADIX
                        03    02EA          .BYTE   RDX$V_HEX
                              02EB
                              02EB  SET_RADIX:
                              02EB
             0000'CF    9A    02EB          MOVZBL  W^MAC$GB_RDXNDX,-           ;SAVE CURRENT INDEX
             0000'CF          02EF                          @^MAC$GL_VALUE
    0000'CF      9E     90    02F2          MOVB    @(SP)+,W^MAC$GB_RDXNDX      ;SET NEW RADIX AND CLEAN STACK
                        05    02F7          RSB                                 ;RETURN TO CALLER'S CALLER
                              02F8
```

```
                           02F8          .SBTTL  OPERATORS
                           02F8
                           02F8  ;++
                           02F8  ; FUNCTIONAL DESCRIPTION:
                           02F8  ;
                           02F8  ;       THESE OPERATOR ROUTINES ARE CALLED WHEN A BINARY OPERATOR
                           02F8  ;       IS ENCOUNTERED IN THE TEXT.  THESE ROUTINES MERELY SET
                           02F8  ;       THE OPERATOR NUMBER INTO MAC$GL_VALUE FOR LATER PROCESSING
                           02F8  ;       BY THE EXPRESSION EVALUATION ROUTINE (EXPBIN).
                           02F8  ;
                           02F8  ;--
                           02F8
                           02F8
                           02F8
                           02F8          .MACRO  OP      OPR
                           02F8          BSBB    SET_UP_OPERATOR
                           02F8          .BYTE   INT$_'OPR
                           02F8          .ENDM
                           02F8
                           02F8
                           02F8  OPPLUS::                                ;OPBINARY = DDPLUS
                           02F8          OP      ADD
                           02FB  OPMINU::                                ;OPBINARY = DDMINUS
                           02FB          OP      SUB
                           02FE  OPMUL::                                 ;OPBINARY = DDTIMES
                           02FE          OP      MUL
                           0301  OPDIV::                                 ;OPBINARY = DDDIV
                           0301          OP      DIV
                           0304  OPAND::                                 ;OPBINARY = DDAND
                           0304          OP      AND
                           0307  OPOR::                                  ;OPBINARY = DDOR
                           0307          OP      OR
                           030A  OPXOR::                                 ;OPBINARY = DDXOR
                           030A          OP      XOR
                           030D  OPASH::                                 ;OPBINARY = DDASH
                           030D          OP      ASH
                           0310  OPCOM::                                 ;OPBINARY = DDUPC
                           0310          OP      NOT
                           0313  OPNEG::                                 ;OPBINARY = DDMINUS
                           0313          OP      NEG
                           0316  OPSAME::                                ;OPBINARY = DDPLUS
                           0316          OP      SAME
                           0319
                           0319  SET_UP_OPERATOR:
   0000'CF    9E     9A    0319          MOVZBL  @(SP)+,W^MAC$GL_VALUE   ;GET THE OPERATOR NUMBER
                     05    031E          RSB                             ;RETURN TO CALLER'S CALLER
```

```
                         031F            .SBTTL  SYMBOL ATTRIBUTE DIRECTIVES -GLOBL/DEBUG/WEAK/EXTRN
                         031F
                         031F     ;++
                         031F     ; FUNCTIONAL DESCRIPTION:
                         031F     ;
                         031F     ;       GLOBAL/DEBUG/WEAK/EXTRN ARE CALLED WHEN THE CORRESPONDING
                         031F     ;       DIRECTIVE IS SCANNED.  FLAGS ARE SET IN SYM_FLAG FOR THE
                         031F     ;       ROUTINE 'IDLIST'.  'IDLIST' IS CALLED FOR EACH SYMBOL IN
                         031F     ;       THE LIST AND IT SETS THE BITS IN SYM_FLAG IN THE SYMBOL
                         031F     ;       BLOCK FOR THAT SYMBOL.
                         031F     ;
                         031F     ;--
                         031F
                         031F     GLOBAL::                                 ;ID_LIST_HEAD = KGLOBL
                 0E   10 031F             BSBB    SET_SYM_FLAG             ;SET FLAG TO REMEMBER
                    0004 0321             .WORD   SYM$M_GLOBL
                         0323
                         0323     DEBUG::                                  ;ID_LIST_HEAD = KDEBUG
                 0A   10 0323             BSBB    SET_SYM_FLAG             ;SET FLAGS TO REMEMBER
                    00A0 0325             .WORD   SYM$M_DEBUG!SYM$M_REF
                         0327
                         0327     WEAK::                                   ;ID_LIST_HEAD = KWEAK
                 06   10 0327             BSBB    SET_SYM_FLAG             ;SET FLAGS TO REMEMBER
                    0006 0329             .WORD   SYM$M_WEAK!SYM$M_GLOBL
                         032B
                         032B     EXTRN::                                  ;ID_LIST_HEAD = KEXTRN
                 02   10 032B             BSBB    SET_SYM_FLAG             ;SET THE FLAG
                    0008 032D             .WORD   SYM$M_EXTRN
                         032F
                         032F
                         032F     SET_SYM_FLAG:
        0000'CF    9E B0 032F             MOVW    @(SP)+,W^SYM_FLAG        ;REMEMBER THE FLAG BIT
                      05 0334             RSB                              ;RETURN TO CALLER'S CALLER
                         0335
                         0335     ;++
                         0335     ; FUNCTIONAL DESCRIPTION:
                         0335     ;
                         0335     ;       AFTER A GLOBAL/DEBUG/WEAK/EXTRN DIRECTIVE HAS BEEN SCANNED,
                         0335     ;       'IDLIST' IS CALLED FOR EACH SYMBOL IN THE LIST OF SYMBOLS
                         0335     ;       ACCOMPANYING THE DIRECTIVE.  THE FLAGS CONTAINED IN SYM_FLAG
                         0335     ;       ARE SET FOR THE SYMBOL.  IF THE DIRECTIVE IS .EXTRN AND
                         0335     ;       THE SYMBOL IS ALREADY DEFINED, AN ERROR MESSAGE IS ISSUED
                         0335     ;       TO PASS 2.
                         0335     ;
                         0335     ;--
                         0335
                         0335     IDLIST::                                 ;ID_LIST = ID
    56   0000'CF47    D0 0335             MOVL    W^MAC$AL_VALSTACK[R7],R6;GET POINTER TO SYMBOL BLOCK
 0D 0000'CF     03    E1 033B             BBC     #SYM$V_EXTRN,W^SYM_FLAG,10$ ;BRANCH IF NOT .EXTRN
   08 09 A6     00    E1 0341             BBC     #SYM$V_DEF,SYM$W_FLAG(R6),10$ ;BRANCH IF SYMBOL NOT DEFINED
                         0346             $MAC_ERR SYMDEFINMO              ; Yes--get error message
              FCB2'   30 034B             BSBW    MAC$ERRORPT              ;SYMBOL DECLARED EXTERNAL BUT ALREADY DEFINE
 09 A6    0000'CF     A8 034E     10$:    BISW    W^SYM_FLAG,SYM$W_FLAG(R6) ;SET BIT(S) IN SYMBOL FLAGS
   05 09 A6     05    E1 0354             BBC     #SYM$V_DEBUG,SYM$W_FLAG(R6),20$ ;BRANCH IF NOT .DEBUG
   00 09 A6     0E    E4 0359             BBSC    #SYM$V_SUPR,SYM$W_FLAG(R6),.+1 ;.DEBUG--CLEAR SUPR BIT
        55   00'8F    9A 035E     20$:    MOVZBL  #CRF$K_REF,R5            ;SET REFERENCE
 50   00000000'EF     D0 0362             MOVL    MAC$GL_PSECTPTR,R0       ;GET POINTER TO PSECT DATA
```

```
               06 0D A0    03   E1  0369    689          BBC     #PSC$V_REL, -              ;IF ABS PSECT
                                    036E    690                  PSC$W_OPTIONS(R0),30$      ;THEN SKIP
             09 A6    0800 8F   A8  036E    691          BISW2   #SYM$M_RELPSECT,SYM$W_FLAG(R6)   ;SET REL PSECT FLAG
                         FC89'  31  0374    692 30$:     BRW     MAC$CREF_SYM               ;CREF SYMBOL IF CREFFING AND RETURN
                                    0377    693
                                    0377    694          .END
```

```
MAC$ACTPRI                     PRIMARIES                      I  4      16-SEP-1984 02:00:18  VAX/VMS Macro V04-00          Page  19
Symbol table                                                             5-SEP-1984 01:47:04  [MACRO.SRC]ACTPRI.MAR;1             (12)

$COUNT         = 0000003B             FLG$M_MOREINP  = 00000008             FLG$V_OPNDCHK  = 00000028
ARG$K_SIZE     = 000003E8             FLG$M_NEWPND   = 00000400             FLG$V_OPRND    = 0000000D
AUD$K_SIZE     = 00000010             FLG$M_NOREF    = 01000000             FLG$V_OPTVFLIDX= 0000002C
BLNK           = 00000020             FLG$M_NTYPEPC  = 00000020             FLG$V_ORDLST   = 00000011
CHR$M_COMMA_CR = 00000020             FLG$M_NULCHR   = 00040000             FLG$V_P2       = 0000000E
CHR$M_ILL_CHR  = 00000040             FLG$M_OBJXST   = 00200000             FLG$V_RPTIRP   = 0000001C
CHR$M_NUM_BER  = 00000010             FLG$M_OPNDCHK  = 00000100             FLG$V_SEQFIL   = 00000019
CHR$M_SPA_MSK  = 00000001             FLG$M_OPRND    = 00002000             FLG$V_SKAN     = 0000000F
CHR$M_SYM_CH1  = 00000008             FLG$M_OPTVFLIDX= 00001000             FLG$V_SPECOP   = 00000022
CHR$M_SYM_CHR  = 00000004             FLG$M_ORDLST   = 00020000             FLG$V_SPLALL   = 0000001A
CHR$M_SYM_DLM  = 00000002             FLG$M_P2       = 00004000             FLG$V_STOIMF   = 00000012
CHR$V_COMMA_CR = 00000005             FLG$M_RPTIRP   = 10000000             FLG$V_SYM2COL  = 0000002A
CHR$V_CVTLWC   = 00000061             FLG$M_SEQFIL   = 02000000             FLG$V_TOCFLG   = 00000013
CHR$V_ILL_CHR  = 00000006             FLG$M_SKAN     = 00008000             FLG$V_UPAFLG   = 00000024
CHR$V_NOCVT    = 0000007F             FLG$M_SPECOP   = 00000004             FLG$V_UPDFIL   = 00000027
CHR$V_NUM_BER  = 00000004             FLG$M_SPLALL   = 04000000             FLG$V_UPMARG   = 00000026
CHR$V_SPA_MSK  = 00000000             FLG$M_STOIMF   = 00040000             FLG$V_XCRF     = 0000001F
CHR$V_SYM_CH1  = 00000003             FLG$M_SYM2COL  = 00000400             GLOBAL           0000031F RG    05
CHR$V_SYM_CHR  = 00000002             FLG$M_TOCFLG   = 00080000             HASHSZ         = 0000007F
CHR$V_SYM_DLM  = 00000001             FLG$M_UPAFLG   = 00000010             HYPHEN         = 0000002D
CNT            = 00000002             FLG$M_UPDFIL   = 00000080             IDLIST           00000335 RG    05
CR             = 0000000D             FLG$M_UPMARG   = 00000040             INP$K_BUFSIZ   = 000003E8
CRF$K_REF        ********  X    05    FLG$M_XCRF     = 80000000             INT$K_BUFSIZ   = 000013F4
DEBUG            00000323 RG    05    FLG$V_ALLCHR   = 00000000             INT$K_BUFWRN   = 00001390
ENTRY_MASK       00000002 R     04    FLG$V_BOL      = 00000001             INT$_ADD       = 00000001
ERR            = 00000001             FLG$V_CHKLPND  = 00000014             INT$_AND       = 00000002
EXPBIN           000001AF RG    05    FLG$V_COMPEXPR = 00000002             INT$_ASH       = 00000003
EXTRN            0000032B RG    05    FLG$V_CONT     = 00000003             INT$_ASN       = 0000000C
FF             = 0000000C             FLG$V_CRF      = 0000001E             INT$_AUGPC     = 0000000D
FLG$M_ALLCHR   = 00000001             FLG$V_CRSEEN   = 00000020             INT$_BDST      = 0000000E
FLG$M_BOL      = 00000002             FLG$V_DATRPT   = 00000004             INT$_CHKL      = 0000000F
FLG$M_CHKLPND  = 00100000             FLG$V_DBGOUT   = 0000002E             INT$_DIV       = 00000004
FLG$M_COMPEXPR = 00000004             FLG$V_DLIMSTR  = 0000002F             INT$_END       = 00000010
FLG$M_CONT     = 00000008             FLG$V_ENDMCH   = 00000005             INT$_EPT       = 00000011
FLG$M_CRF      = 40000000             FLG$V_EVALEXPR = 00000006             INT$_ERR       = 00000012
FLG$M_CRSEEN   = 00000001             FLG$V_EXPOPT   = 00000007             INT$_ETX       = 00000013
FLG$M_DATRPT   = 00000010             FLG$V_EXTERR   = 00000030             INT$_FNEWL     = 00000014
FLG$M_DBGOUT   = 00004000             FLG$V_EXTWRN   = 00000031             INT$_ILG       = 00000000
FLG$M_DLIMSTR  = 00008000             FLG$V_FIRSTLN  = 00000029             INT$_INFO      = 0000003A
FLG$M_ENDMCH   = 00000020             FLG$V_IFSTAT   = 00000017             INT$_LGLAB     = 00000015
FLG$M_EVALEXPR = 00000040             FLG$V_IIF      = 00000016             INT$_MACL      = 00000016
FLG$M_EXPOPT   = 00000080             FLG$V_INSERT   = 00000008             INT$_MUL       = 00000005
FLG$M_EXTERR   = 00010000             FLG$V_IRPC     = 0000001D             INT$_NEG       = 00000006
FLG$M_EXTWRN   = 00020000             FLG$V_LEXOP    = 00000021             INT$_NEWL      = 00000017
FLG$M_FIRSTLN  = 00000200             FLG$V_LSTXST   = 00000009             INT$_NEWP      = 00000018
FLG$M_IFSTAT   = 00800000             FLG$V_MAC2COL  = 0000002B             INT$_NOT       = 00000007
FLG$M_IIF      = 00400000             FLG$V_MACL     = 0000000B             INT$_OP        = 00000019
FLG$M_INSERT   = 00000100             FLG$V_MACLTB   = 0000001B             INT$_OR        = 00000008
FLG$M_IRPC     = 20000000             FLG$V_MACTXT   = 00000010             INT$_PRIL      = 0000001A
FLG$M_LEXOP    = 00000002             FLG$V_MEBLST   = 0000000C             INT$_PRT       = 0000001B
FLG$M_LSTXST   = 00000200             FLG$V_MOREARG  = 0000002D             INT$_PSECT     = 0000001C
FLG$M_MAC2COL  = 00000800             FLG$V_MOREINP  = 00000023             INT$_REDEF     = 0000001D
FLG$M_MACL     = 00000800             FLG$V_NEWPND   = 0000000A             INT$_REF       = 0000001E
FLG$M_MACLTB   = 08000000             FLG$V_NOREF    = 00000018             INT$_REST      = 0000001F
FLG$M_MACTXT   = 00010000             FLG$V_NTYPEPC  = 00000025             INT$_SAME      = 00000009
FLG$M_MEBLST   = 00001000             FLG$V_NULCHR   = 00000032             INT$_SAVE      = 00000020
FLG$M_MOREARG  = 00002000             FLG$V_OBJXST   = 00000015             INT$_SBTTL     = 00000021
```

```
                                                          J  4
MAC$ACTPRI                         PRIMARIES                                  16-SEP-1984 02:00:18  VAX/VMS Macro V04-00        Page  20
Symbol table                                                                   5-SEP-1984 01:47:04  [MACRO.SRC]ACTPRI.MAR;1          (12)

INT$_SETFLAG    = 00000022                    MAC$_UNTERMARG = 007D922A                   PSC$M_LCL       = FFFFFFEF
INT$_SETLONG    = 00000023                    MAC_SUBSYS     = 0000007D                   PSC$M_LIB       = 00000002
INT$_SPIC       = 00000024                    MASK             0000017F RG     05          PSC$M_LONG      = 00004800
INT$_SPID       = 00000025                    MASKNL           00000194 RG     05          PSC$M_NOEXE     = FFFFFFBF
INT$_STIB       = 00000026                    MASKX            00000186 RG     05          PSC$M_NOPIC     = FFFFFFFE
INT$_STIL       = 00000028                    NUMASC           0000025F RG     05          PSC$M_NORD      = FFFFFF7F
INT$_STIW       = 00000027                    NUMFLT           00000100 RG     05          PSC$M_NOSHR     = FFFFFFDF
INT$_STKEPT     = 00000029                    OBJ$K_BUFSIZ   = 00000200                   PSC$M_NOVEC     = FFFFFDFF
INT$_STKG       = 0000002A                    OPAND            00000304 RG     05          PSC$M_NOWRT     = FFFFFEFF
INT$_STKL       = 0000002B                    OPASH            0000030D RG     05          PSC$M_OVR       = 00000004
INT$_STKPC      = 0000002C                    OPCOM            00000310 RG     05          PSC$M_PAGE      = 00006400
INT$_STKS       = 0000002D                    OPDIV            00000301 RG     05          PSC$M_PIC       = 00000001
INT$_STOB       = 00000034                    OPF$M_LASTOPR  = 00002000                   PSC$M_QUAD      = 00004C00
INT$_STOL       = 0000002E                    OPF$M_OPTEXP   = 00001000                   PSC$M_RD        = 00000080
INT$_STOW       = 00000035                    OPF$V_LASTOPR  = 0000000D                   PSC$M_REL       = 00000008
INT$_STRB       = 0000002F                    OPF$V_OPTEXP   = 0000000C                   PSC$M_SHR       = 00000020
INT$_STRL       = 00000031                    OPMINU           000002FB RG     05          PSC$M_USR       = FFFFFFFD
INT$_STRSB      = 00000032                    OPMUL            000002FE RG     05          PSC$M_VEC       = 00000200
INT$_STRSW      = 00000033                    OPNEG            00000313 RG     05          PSC$M_WORD      = 00004400
INT$_STRW       = 00000030                    OPOR             00000307 RG     05          PSC$M_WRT       = 00000180
INT$_STSB       = 00000036                    OPPLUS           000002F8 RG     05          PSC$S_ALIGNMENT= 00000004
INT$_STSW       = 00000037                    OPSAME           00000316 RG     05          PSC$V_ALIGNFLG  = 0000000E
INT$_SUB        = 0000000A                    OPXOR            0000030A RG     05          PSC$V_ALIGNMENT= 0000000A
INT$_SUME       = 00000039                    P1$ARITH_ADD     ********    X   03          PSC$V_EXE       = 00000006
INT$_WRN        = 00000038                    P1$ARITH_AND     ********    X   03          PSC$V_GBL       = 00000004
INT$_XOR        = 0000000B                    P1$ARITH_ASH     ********    X   03          PSC$V_LIB       = 00000001
LST$K_BUFSIZ    = 00000086                    P1$ARITH_DISP    00000000 RG     03          PSC$V_OVR       = 00000002
LST$K_L_P_PAGE  = 0000003C                    P1$ARITH_DIV     ********    X   03          PSC$V_PIC       = 00000000
LST$K_TITLE_SIZ= 00000028                     P1$ARITH_MUL     ********    X   03          PSC$V_RD        = 00000007
MAC$AL_VALSTACK   ********   X     05         P1$ARITH_NEG     ********    X   03          PSC$V_REL       = 00000003
MAC$CREF_SYM      ********   X     05         P1$ARITH_NOT     ********    X   03          PSC$V_SHR       = 00000005
MAC$ERRORPT       ********   X     05         P1$ARITH_OR      ********    X   03          PSC$V_VEC       = 00000009
MAC$GB_RDXNDX     ********   X     05         P1$ARITH_SAME    ********    X   03          PSC$V_WRT       = 00000008
MAC$GETCHR        ********   X     05         P1$ARITH_SUB     ********    X   03          PSC$W_FLAG        00000009
MAC$GETFLOAT      ********   X     05         P1$ARITH_XOR     ********    X   03          PSC$W_OPTIONS     0000000D
MAC$GL_ABSFLAG    ********   X     05         PRMBRK           0000011C RG     05          RDX$V_BINARY    = 00000000
MAC$GL_ERRPT      ********   X     05         PRMINT           00000108 RG     05          RDX$V_DECIMAL   = 00000002
MAC$GL_EXPEND     ********   X     05         PRMPC            0000012E RG     05          RDX$V_DOUBLE    = 00000005
MAC$GL_HIGH_32    ********   X     05         PRMRDX           00000125 RG     05          RDX$V_FLOAT     = 00000004
MAC$GL_OPSIZE     ********   X     05         PRMSYM           00000063 RG     05          RDX$V_GFLOAT    = 00000006
MAC$GL_PC         ********   X     05         PRMUN            00000000 RG     05          RDX$V_HEX       = 00000003
MAC$GL_PRMSEG     ********   X     05         PSC$B_NAME       00000004                   RDX$V_HFLOAT    = 00000007
MAC$GL_PSECT      ********   X     05         PSC$B_SEG        0000000C                   RDX$V_OCTAL     = 00000001
MAC$GL_PSECTPTR   ********   X     05         PSC$B_UNUSED     0000000B                   RDXBIN            000002DF RG     05
MAC$GL_VAL3       ********   X     05         PSC$K_BLKSIZ     00000013                   RDXDEC            000002E2 RG     05
MAC$GL_VALUE      ********   X     05         PSC$K_NO_OPTNS = 0000000A                   RDXHEX            000002E8 RG     05
MAC$GQ_HIGH_64    ********   X     05         PSC$L_CURLOC     0000000F                   RDXOCT            000002E5 RG     05
MAC$GQ_VAL2       ********   X     05         PSC$L_LINK       00000000                   REG$_PC         = 0000000F
MAC$GQ_VALUEQ     ********   X     05         PSC$L_MAXLGTH    00000005                   REGLST            00000172 RG     05
MAC$INTERR_2_LW   ********   X     05         PSC$M_ABS      = FFFFFFF7                   RGLST1            0000016E RG     05
MAC$INTOUT_1_LW   ********   X     05         PSC$M_ALIGNFLG = 00004000                   SEMI            = 0000003B
MAC$INTOUT_X      ********   X     05         PSC$M_ALLOPTNS = 000003FF                   SET_RADIX         000002EB R      05
MAC$SKIPSP        ********   X     05         PSC$M_BYTE     = 00004000                   SET_SYM_FLAG      0000032F R      05
MAC$_DATATRUNC = 007D8800                     PSC$M_CON      = FFFFFFFB                   SET_UP_OPERATOR   00000319 R      05
MAC$_DIVBYZERO = 007D8808                     PSC$M_DEFAULT  = 000001C8                   SIGN_BIT        = 80000000
MAC$_EXPOVR32  = 007D8810                     PSC$M_EXE      = 000000C0                   STB$K_PG_MISS   = 0000000A
MAC$_SYMDEFINMO= 007D91E2                     PSC$M_GBL      = 00000010                   SYM$B_NAME        00000004
```

```
SYM$B_SEG            0000000C
SYM$B_TOKEN          0000000B
SYM$K_BLKSIZ         0000000D
SYM$K_MAXLEN       = 0000001F
SYM$K_TWOCOL       = 00000010
SYM$L_LINK           00000000
SYM$L_VAL            00000005
SYM$M_ABS          = 00000010
SYM$M_ASN          = 00000100
SYM$M_CRFO         = 00002000
SYM$M_DEBUG        = 00000020
SYM$M_DEF          = 00000001
SYM$M_DELMAC       = 00000200
SYM$M_EPT          = 00000200
SYM$M_EXTRN        = 00000008
SYM$M_GLOBL        = 00000004
SYM$M_LOCAL        = 00000040
SYM$M_ODBG         = 00000400
SYM$M_REF          = 00000080
SYM$M_RELPSECT     = 00000800
SYM$M_SUPR         = 00004000
SYM$M_WEAK         = 00000002
SYM$M_XCRF         = 00001000
SYM$V_ABS          = 00000004
SYM$V_ASN          = 00000008
SYM$V_CRFO         = 0000000D
SYM$V_DEBUG        = 00000005
SYM$V_DEF          = 00000000
SYM$V_DELMAC       = 00000009
SYM$V_EPT          = 00000009
SYM$V_EXTRN        = 00000003
SYM$V_GLOBL        = 00000002
SYM$V_LOCAL        = 00000006
SYM$V_ODBG         = 0000000A
SYM$V_REF          = 00000007
SYM$V_RELPSECT     = 0000000B
SYM$V_SUPR         = 0000000E
SYM$V_WEAK         = 00000001
SYM$V_XCRF         = 0000000C
SYM$W_FLAG           00000009
SYM_FLAG             00000000 R       04
TAB                = 00000009
WEAK                 00000327 RG      05
X1                 = 00000033
X2                 = 00080000


                                             +-----------------+
                                             ! Psect synopsis !
                                             +-----------------+

PSECT name                         Allocation          PSECT No.  Attributes
----------                         ----------          ---------  ----------
.  ABS  .                          00000000  (    0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
. BLANK .                          00000000  (    0.)  01 (  1.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$ABS$                              00000013  (   19.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
MAC$RO_DATA                        00000030  (   48.)  03 (  3.)  NOPIC    USR    CON    REL    GBL NOSHR NOEXE   RD   NOWRT NOVEC LONG
MAC$ACTPRI_DATA                    00000004  (    4.)  04 (  4.)  NOPIC    USR    CON    REL    LCL NOSHR NOEXE   RD     WRT NOVEC LONG
```

```
MAC$RO_CODE_P1                      00000377  (  887.)  05 (  5.)  NOPIC   USR   CON   REL   GBL NOSHR   EXE   RD  NOWRT NOVEC LONG

                                                   +--------------------------+
                                                   ! Performance indicators !
                                                   +--------------------------+

Phase                        Page faults   CPU Time       Elapsed Time
-----                        -----------   --------       ------------
Initialization                        35   00:00:00.03    00:00:02.43
Command processing                   131   00:00:00.37    00:00:03.59
Pass 1                               216   00:00:03.53    00:00:15.39
Symbol table sort                      0   00:00:00.43    00:00:00.97
Pass 2                               135   00:00:01.18    00:00:02.71
Symbol table output                   32   00:00:00.15    00:00:00.32
Psect synopsis output                  2   00:00:00.02    00:00:00.02
Cross-reference output                 0   00:00:00.00    00:00:00.00
Assembler run totals                 553   00:00:05.72    00:00:25.43

The working set limit was 1500 pages.
34438 bytes (68 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 462 non-local and 33 local symbols.
694 source lines were read in Pass 1, producing 25 object records in Pass 2.
16 pages of virtual memory were used to define 15 macros.

                                                  +----------------------------+
                                                  ! Macro library statistics !
                                                  +----------------------------+

Macro library name                                Macros defined
------------------                                --------------
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1                          12
_$255$DUA28:[SYSLIB]STARLET.MLB;2                            3
TOTALS (all libraries)                                      15

506 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:ACTPRI/OBJ=OBJ$:ACTPRI MSRC$:ACTPRI/UPDATE=(ENH$:ACTPRI)+LIB$:MACRO/LIB
```

0224 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

ACTPRI
LIS

ARGSCN
LIS

BDYSCN
LIS

CRFSUB
LIS

ACTOPC
LIS

ACTSTA
LIS

APSECT
LIS

CRFDAT
LIS

ACTREF
LIS

COMPUT
LIS